REMARKS

Claims 1-69 remain in the application for consideration. In view of the following amendments and remarks, Applicant respectfully solicits allowance of the application and furtherance onto issuance.

Drawing Objection

The drawings are objected to as containing a reference number that does not appear in the specification. Applicant has amended the specification as set forth below to correct this deficiency. Applicant thanks the Examiner for the Examiner's attention to detail.

Specification Objection

The Specification has been objected to for a couple of different reasons. First, the Office notes the that table appearing on page 21 appears to have some words missing. Applicant has reviewed the subject table and submits that no words are missing. Specifically, the Office will note that there are a number of Group Names that appear in the table under the heading "Group names". A column designated "When downloaded" describes when particular files are downloaded with respect to the individual groups referenced in the Group name column. Specifically, for the Group name "Required", the "When downloaded" column indicates "Downloaded before any other files in the extension". Similarly, for the Group name "Offline", the "When downloaded" column indicates "Offline files start getting downloaded as soon as Required are down." The "Required" to which this description refers is the "Required" group appearing just above in the table. Accordingly, no words are missing. The language in the table is simply referring back to a previous Group.

Second, the Office notes that there appears to be an informality on page 35, line 5, insofar as the language "set up and..." is concerned. The excerpt of the specification referenced by the Office refers to a flow chart that describes steps in a process that can be referred to as a "set up and extension installation process." That is, the steps include both "set up" steps and "installation" steps. Thus, there is no informality.

Applicant does, however, sincerely appreciate the Office's attention to detail in reading the specification to ensure there are no mistakes.

§ 101 Rejections

Claims 40-47 stand rejected under 35 U.S.C. § 101 as being directed to non-statutory subject matter. Specifically, the Office argues that the claimed subject matter is directed to a data structure and is hence non-statutory.

Claim 40 recites a data structure embodied on a computer-readable medium comprising:

- a first sub-structure indicative of a software extension that is to be incorporated in a software application program;
- one or more second sub-structures associated with the first sub-structure and indicative of feature types that can be added by the extension to the application program; and
- one or more third sub-structures associated with the one or more second sub-structures and indicative of features of an associated feature type that can be added by the extension.

In making out the rejection, the Office notes, as between the different types of "descriptive material" there is "functional descriptive material" and "non-functional descriptive material". In this context, as noted by the Office, "functional descriptive material" consists of ***data structures*** and computer programs which impart functionality when employed as a computer component. Further, as noted by the Office, a data structure is defined as "a physical or logical relationship among data elements, designed to support specific data manipulation functions."

As further noted by the Office, data structures are in most cases statutory when interrelated to a medium, such as a computer-readable medium.

In the present case, claim 40 is clearly directed to functional descriptive material. Specifically, as set forth in the preamble of the claim, claim 40 is directed to "a data structure embodied on a computer-readable medium."

Applicant's disclosure describes methods and systems for providing software via a network. See, page 1, lines 21-24. One aspect of the disclosure is directed to a data structure that describes software extensions, feature types that can be added by the extensions, and features of an associated feature type that can be added by the extension. The claimed data structure can be utilized to download software that is provided via a network. Thus, the claimed data structure facilitates delivery and downloading of the software extensions. As an example, see the Specification, page 12-16 which describes but one example of a data structure aspects of which are embodied in claim 40.

To this extent, the claimed data structure is a "physical or logical relationship among data elements" (i.e. the first, second and third sub-structures) "designed to support specific data manipulation function" (i.e. delivery of software extensions via a network).

Accordingly, taken in the context of Applicant's specification, claim 40 and its dependent claims 41-47, which recite functional descriptive material embodied on a computer-readable medium, recite statutory subject matter. Applicant respectfully solicits withdrawal of this rejection.

§ 112 Rejections

Claims 64 and 65 stand rejected under 35 U.S.C. § 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which Applicant regards as the invention. Specifically, the Office argues that the term "A network site" is not descriptive enough. Applicant respectfully disagrees, particularly when the claim is read in light of the specification. Specifically, claim 64 recites a ***network site*** through which a client can access software files comprising:

- one or more software extension files configured to be incorporated into a software application program and delivered via a network; and
- one or more files associated with the one or more software extension files and describing the extension files, the one or more files describing a logical attachment of the one or more software extension files to the software application program.

The term "site" is discussed in the application in a manner that renders a clear and unambiguous meaning as to what is specifically claimed in this claim. For example, page 6, lines 3-10 states the following:

> The methods and systems described just below provide a mechanism by which functionality can be added dynamically to an application program or software platform. Functionalities or "extensions" as they will be referred to, can be advantageously added via a network such as the Internet. Extensions, that can implement new features or add to existing features, can be added using only a network address, such as a URL, as a basis for extension installation. *That is, all of the files that comprise an extension can be maintained on the network and accessed via one or more network **sites.***

That is, a "site" as used in this claim, pertains to a location that is associated with a network address, such as a URL. Applicant submits that this is sufficiently clear as to particularly point out and distinctly claim the subject matter which Applicant regards as its own. Accordingly, Applicant respectfully solicits withdrawal of the rejection.

§ 102 Rejections

Claims 1-69 stand rejected under 35 U.S.C § 102(e), as being anticipated by U.S. Patent No. 6,253,366 to Mutschler, III (hereinafter "Mutschler").

Before discussing the substance of the Office's rejection of the above-mentioned claims, the following discussion of Mutschler and Applicant's disclosure is provided to assist the Office in appreciating the patentable distinctions between Applicant's claimed embodiments and the cited references.

The Mutschler Reference

Mutschler is directed to methods and systems for generating a compact Document Type Definition (DTD) for data interchange among software tools. The methods and systems described by Mutschler make use of XML.

Mutschler's methods and systems are utilized in connection with repositories, which provide a central place for recording ***metadata*** and enable one to store, manage, share and reuse information about data (i.e., metadata) that an enterprise uses. As noted by Mutschler, a repository can store definitional, management and operational information. Tools can be integrated with the repository to support information sharing and metadata reuse and tool and technology models may be developed to manipulate the tool information in the repository. However, the transferring of data within models from tool to tool or from a tool to the repository has been a cumbersome and unyielding task for a long time. This is precisely the problem that Mutschler sets out to address with its invention. Preliminarily, please note that this has nothing to do with methods and systems for delivering software extensions.

Repository models, as described by Mutschler, typically contain classes, datatypes and messages. As more and more complex models are being built, the need arises for a method and system to transfer data in a model from place to place, e.g., to a tool that understands the UML ("Universal Modeling Language"). Mutschler's disclosure purportedly solves this problem by generating a data-transfer syntax in which a tool using a meta-model can transport data from place to place.

The prefix "meta", as used by Mutschler, describes a relationship. For example, "meta-data" describes data. In a similar fashion, a meta-object is an object that represents "meta-data"; and, "meta-model" means a model that defines an abstract language for expressing other models.

As noted by Mutschler, it is a tedious and time consuming task to generate a format description for enabling the interchange of metadata among repositories and each different type of modeling tool available. Accordingly, Mutschler addresses a need for automatically generating format descriptions to expedite interchange of metadata among repositories and modeling tools.

Mutschler disclosure describes the invention as "algorithms for generating an XMI DTD for any valid meta-model defined in a MOF-compliant repository." An "XMI" refers to "XML Metadata Interchange", which is an open industry standard that combines the benefits of the Web-based XML standard for defining, validating and sharing document formats on the Web with the Meta Object Framework (MOF) to provide a means for generating ***formats*** to allow the development tools to share information. See, column 4, lines 19-39.

According to Mutschler, in order to accomplish the objects of its invention it is necessary to generate Document Type Definitions ("DTD") for the Extensible Markup Language ("XML"), a World Wide Web Consortium standard. A DTD is a set of rules governing the element types that are allowed within an XML document and rules specifying the allowed content and attributes of each element type. The DTD also declares all the external entities referenced within the document and the notations that can be used. Stated otherwise, an XML DTD provides a means by which an XML processor can validate the syntax and some of the semantics of an XML document. An XMI DTD specifies the particular elements allowed in an XMI document. See, e.g. column 4, lines 47-60.

Mutschler describes its invention in more detail in connection with Fig. 1. There, a block diagram shows a server 10 that executes a variety of software including a repository 11 and object services 12. The repository 11 includes a set of repository services 13, which also couple the repository to an object request broker ("ORB") 14. The object services 12 also couples the server to the ORB 14.

A first tool 15, which is being executed by a first workstation 16, is coupled to the ORB 14. In a similar manner, a second tool 17, which is being executed by a second workstation 18, is also coupled to the ORB 14. A DTD generator 19 is provided and effects data interchange among the tools 15 and 17 and the repository 11 by defining the contents of the messages exchanged. The DTD is first generated then it is subsequently employed for communication by the repository 11 with the tools 15 and 17.

In operation, the DTD generator 19 accesses the meta-model 20 (Fig. 2) and then produces a DTD (bubble 19A). Using the DTD thus created, the Repository Services 13 is able to generate an XMI data stream from the Application Model 21. This XMI data stream can thus be communicated through the ORB 14 to the input/export module 22 or 23. The DTD is then used by the module 22 or 23 to place the data from the Application Model 21 into the Tool 15 or the Tool 17.

Nowhere does Mutschler disclose or suggest software extensions or the delivery thereof as contemplated in Applicant's disclosure.

Applicant's Disclosure

Applicant's disclosure is directed to providing new ***software delivery models*** that are particularly well-suited for network-based software delivery, e.g. delivery via the Internet. See, Specification, page 2, lines 1-22.

As noted above, Mutschler is not directed to providing methods and systems for ***delivering*** software, as contemplated in Applicant's disclosure. Once these fundamental differences are appreciated, the patentable distinctions between

Applicant's claimed subject matter and the cited reference are more easily appreciated.

As noted in Applicant's "Overview" section starting on page 6, line 1, Applicant's methods and systems provide a mechanism by which functionality can be added dynamically to an application program or software platform. Functionalities or extensions can be advantageously added via a network such as the Internet. Extensions, that can implement new features or add to existing features, can be added using only a network address, such as a URL, as a basis for extension installation. That is, all of the files that comprise an extension can be maintained on the network and accessed via one or more network sites.

Extensions can be described in a variety of ways. One way utilizes a hierarchical tag-based language which facilitates handling and use of the various extensions. In one particular implementation, a software platform is provided that can incorporate various functionalities. The software platform and the inventive methods and systems enable third and fourth party developers to develop extensions for the platform that can be easily and seamlessly incorporated into the platform without having any knowledge of (or relationship with) a hosting service. A third party developer is a developer who develops an extension for the platform. A fourth party developer might be a developer who develops an extension to a third party developer's extension. Thus, the incorporation of third and fourth party extensions is essentially a transparent process, as far as developers are concerned.

Consider for example, Applicant's Fig. 1, which shows a user's computer 100 and several so-called extension sources 102, 104, and 106. The extension sources can comprise any entity from which a software extension can be obtained via a network. In an exemplary implementation, the network can comprise the

Internet, although other networks (e.g. LANs and WANs) can certainly be utilized. Extension sources can include, without limitation, business entities such as retail stores that might maintain a network site. In one implementation, a user can execute software on their computer that provides an application program or software platform. Each of the different extension sources 102-106 can provide software extensions that can plug into the software platform that is executing on the user's machine. These extensions are deliverable via a network such as the Internet, and assist in providing applications that can be executed on the user's machine. In some embodiments, the extensions are logically described in XML. Additionally, the use of XML assists in the future discoverability of extensions by promoting XML DOM properties on the Internet. It will be appreciated, however, that any suitable format can be used for describing the extensions, e.g. a binary description could be used.

Extensions can be delivered from any number of different extension sources. The inventive methods and systems provide a streamlined and organized way to handle the provided extensions. The use of XML advantageously enables efficient handling of extensions from multiple different extension sources, without unduly taxing the software components that utilize specific portions of an extension or extensions.

In the embodiment in the Specification, extensions are organized in three separate but related portions: an Extension Definition File (EDF), a Package Manifest (PKG), and the code, components, or "bits" that make up or define the extension. An EDF can be, but need not be associated with a URL (Universal Resource Locator) that provides a way for a client to access the EDF. By convention and choice, the PKG file is located at the same URL as the EDF.

EDFs describe logical attachments to an application program or software platform, while PKGs specify the physical files and resources that are used in an extension. There can be a one to one correspondence between EDFs and PKGs.

Fig. 3 shows an exemplary organization 300 that includes an EDF 302 and a corresponding package manifest (PKG) 304. In the illustrated example, the EDF 302 uses XML to describe the logical attachments or extensions to an application program. The corresponding PKG 304 specifies the physical files and resources that are associated with a particular extension.

In accordance with one embodiment, an EDF is an XML file that logically describes an extension. For example, the EDF can describe HTML that makes up a user interface (UI), the objects that contain code for implementing various functions, and the like. The EDF can also contain all or part of the functionality that comprises an extension. For instance, the HTML that describes a toolbar could be incorporated directly into an EDF file, and a toolbar attachment manager could read it from the EDF file, instead of from a URL. The information contained in the EDF is processed and (together with the information contained in the PKG), the appropriate files are automatically installed on a user's computer. This is done unobtrusively without manipulating the computer's persisted settings, as might be found in the user's system registry.

An EDF, when implemented in XML, contains various tags that are associated with various extensions. The various tags can correspond to:

- User interface elements
- Behaviors/Components/Objects
- Store Elements
- User-defined objects

- Or anything else that represents a point of extensibility in the application or platform

EDFs can also have one or more predefined tags. Exemplary predefined XML tags for user interface elements can include tags for feature types such as: tool bars, accelerators, menu items, and themes. Exemplary predefined XML tags for behaviors/components/objects include tags for Services. Exemplary predefined XML tags for store elements include tags for content classes and offline data sources.

In XML embodiments, EDFs can have a particular XML schema that is utilized. The schema comprises collections of XML tags that are arranged in a hierarchical organization to facilitate information dissemination to software components that need certain extensions.

Top level tags in an EDF can be associated with a feature type that can be added by a particular extension. Underneath each top level tag there can be one or more child tags that are individually associated with a particular feature of the feature type that is to be added by a particular extension.

Fig. 5 shows an exemplary XML schema organization in accordance with one embodiment. For each top level tag in an EDF, there is an associated attachment manager which is a software component that receives data associated with the tag so that the data can be used to incorporate the extension into the platform or application program.

The package manifests (PKGs) assist in organizing the downloading of software in the form of multiple files over a network such as the Internet. The PKGs can be employed with EDFs. While the EDFs describe the logical attachment of extensions into an application program or platform, the package manifest's role is to assist in one or more of: organized delivery, validation and/or, updating of files associated with the various extensions that can be provided.

Thus, the various methods and systems described in Applicant's disclosure are directed to providing methods and systems for delivering software via a network, such as the Internet.

The Claim Rejections

Claim 1 recites a method for ***delivering software via a network*** comprising:

- ***describing one or more software extensions*** using descriptions, the extensions being configured for incorporation in a software platform executing on a client; and
- ***delivering the descriptions of the one or more extensions to the client via the network***, the descriptions being configured for use in downloading the software extensions via the network.

In making out the rejection of this claim, the Office cites to Mutschler's column 4, lines 21-30 and 48-60 (as anticipating the "describing" act), and to column 6, lines 13-16 (as anticipating the "delivering" act). Applicant respectfully disagrees.

The excerpts cited by the Office are set out in their entireties below:

> Users of workgroup-based and component development tools are finding it increasingly difficult to coordinate their software development efforts across the enterprise. A solution in accordance with the present invention employs the benefits of XMI (XML Metadata Interchange), which is an open industry standard that combines the benefits of the Web-based XML standard for defining, validating and sharing document formats on the Web with the Meta Object Framework (MOF) to provide a means for generating formats to allow the development tools to share information.

> One particular use of this invention is to define an XML DTD for the object-oriented Unified Modeling Language (UML). *Column 4, lines 21-30.*
>
> In order to accomplish the objects of the present invention it is necessary to generate Document Type Definitions ("DTD") for the Extensible Markup Language ("XML"), a World Wide Web Consortium standard. A DTD is a set of rules governing the element types that are allowed within an XML document and rules specifying the allowed content and attributes of each element type. The DTD also declares all the external entities referenced within the document and the notations that can be used. Stated otherwise, an XML DTD provides a means by which an XML processor can validate the syntax and some of the semantics of an XML document. An XMI DTD specifies the particular elements allowed in an XMI document. *Column 4, lines 48-60.*
>
> The ORB 14 is coupled to the tool 15 by means of an import/export module 22; and, in a like manner to the tool 17 by means of an import/export module 23. The term "import" as used herein shall mean the creation of an object based on a description of an object transmitted from an external entity. *Column 6, lines 13-16.*

Applicant respectfully submits that Mutschler in general, and these excerpts specifically, have nothing to do with "***describing one or more software extensions*** using descriptions, the extensions being configured for incorporation in a software platform executing on a client", and "***delivering the descriptions of the one or more extensions to the client via the network***, the descriptions being configured for use in downloading the software extensions via the network." Rather, the excerpts cited by the Office simply describe aspects of Mutschler's methods and systems for generating a compact Document Type Definition (DTD) for data interchange among software tools. Mutschler's methods and systems do not disclose or suggest the subject matter of this claim. Accordingly, this claim is allowable.

Claims 2-16 depend from claim 1 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 1, are neither disclosed nor suggested in the references of record, either singly or in combination with one another. In making out the rejections of these claims, the Office cites to various portions of Mutschler. Applicant has thoroughly studied Mutschler and, in particular, the excerpts cited by the Office, and can find no disclosure or suggestion whatsoever of the subject matter of these claims.

Claim 17 recites one or more computer-readable media having computer-readable instructions thereon which, when executed by a computer system, cause the computer system to:

- ***describe one or more software extensions using extensible markup language (XML)***, the extensions being configured for incorporation in a software platform comprising ***a single application program***, the single application program having multiple different functionalities that can enable a user to accomplish multiple different tasks; and
- ***deliver XML descriptions of the one or more extensions to the client via the Internet***, the descriptions being configured for use in downloading the software extensions via the Internet.

In making out the rejection of this claim, the Office cites to the same sections of Mutschler as were cited to in making out the rejection of claim 1. As noted above, these excerpts simply describe aspects of Mutschler's methods and systems for generating a compact Document Type Definition (DTD) for data interchange among software tools. Mutschler generally, and these excerpts specifically, have nothing to do with describing software extensions, and delivering those extension descriptions via the Internet. Accordingly, for at least this reason, this claim is allowable.

Claim 18 recites a method for delivering software via a network comprising:

- ***describing one or more software extensions using one or more descriptive files***, the extensions being configured for incorporation in a software program executing on a client;
- ***associating the one or more descriptive files with one or more associated extension files*** that are useable to provide a program functionality;
- ***storing the descriptive files and associated extension files*** in a network-accessible location; and
- ***delivering the descriptive files and the associated extension files*** of the one or more extensions to the client via a network.

As noted above, Mutschler is not directed to describing and delivering software extensions, as contemplated in Applicant's disclosure. In making out the rejection of this claim, the Office cites to column 4, lines 21-30 and 48-60, and column 6, lines 13-16 (as anticipating the "describing" act), to column 4, lines 54-56 (as anticipating the "associating" act), to Fig. 2, DTD (as anticipating the "storing" act), and to Fig. 1 (as anticipating the "delivering" act). Applicant respectfully disagrees.

The cited passages in Mutschler have nothing to do with describing software extensions, associating descriptive files with one or more extension files that are useable to provide a program functionality, storing the descriptive files and the extension files in a network-accessible location, and delivering the descriptive files and associated extension files. Accordingly, for at least these reasons, this claim is allowable.

Claims 19-28 depend from claim 18 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 18, are neither disclosed nor suggested in the references of record, either singly or in combination with one another. In making out the rejections of these claims, the Office cites to various portions of Mutschler. Applicant has thoroughly studied Mutschler and, in particular, the excerpts cited by the Office, and can find no disclosure or suggestion whatsoever of the subject matter of these claims.

Claim 29 recites a method of delivering software via a network comprising:

- ***storing one or more extension definition files (EDFs)*** that describe a logical attachment to a software application program;
- ***storing one or more extension files*** that correspond to the one or more EDFs and extend the software application program; and
- ***delivering, via a network, at least one EDF*** to a client; and
- ***delivering, via a network, at least one extension file*** that corresponds to the at least one EDF to a client.

In making out the rejection of this claim, the Office cites to Figs. 1 and 2 (as anticipating the first "storing" act), to column 2, lines 37-43 (as anticipating the second "storing" act), and to column 4, lines 29-39 (as anticipating the "delivering" acts). Applicant respectfully disagrees.

Mutschler in no way discloses or suggests the subject matter of this claim. The specifics of Mutschler's disclosure have been pointed out above and, for the sake of brevity, are not repeated. Suffice it to say, however, that Mutschler's systems and methods do not in any way perform the "storing" and "delivering" acts described above. Accordingly, for at least this reason, this claim is allowable.

Claims 30-39 depend from claim 29 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 29, are neither disclosed nor suggested in the references of record, either singly or in combination with one another. In making out the rejections of these claims, the Office cites to various portions of Mutschler. Applicant has thoroughly studied Mutschler and, in particular, the excerpts cited by the Office, and can find no disclosure or suggestion whatsoever of the subject matter of these claims.

Claim 40 recites a data structure embodied on a computer-readable medium comprising:

- a first sub-structure indicative of a software extension that is to be incorporated in a software application program;
- one or more second sub-structures associated with the first sub-structure and indicative of feature types that can be added by the extension to the application program; and
- one or more third sub-structures associated with the one or more second sub-structures and indicative of features of an associated feature type that can be added by the extension.

In making out the rejection of this claim, the Office cites to column 4, lines 1-8, and 21-39 (as anticipating the first, second and third sub-structures). Applicant respectfully disagrees. As noted above, Mutschler does not disclose or suggest any systems or methods that are employable with "extensions" that can be incorporated in a software application program. As this claim recites a data structure that is used in connection with software extensions that can be incorporated in a software application program, it is virtually impossible for

Mutschler to disclose or suggest any such data structure. Accordingly, for at least this reason, this claim is allowable.

Claims 41-47 depend from claim 40 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 40, are neither disclosed nor suggested in the references of record, either singly or in combination with one another. In making out the rejections of these claims, the Office cites to various portions of Mutschler. Applicant has thoroughly studied Mutschler and, in particular, the excerpts cited by the Office, and can find no disclosure or suggestion whatsoever of the subject matter of these claims.

Claim 48 recites a method of ***delivering software via a network*** comprising:

- navigating to a network site that maintains at least one software application program; and
- downloading a software application program from the network site, the application program comprising multiple different functionalities that can assist a user in accomplishing different tasks, the software application program being configured to be extended with software extensions that are deliverable via a network and are described by at least one network-deliverable file.

In making out the rejection of this claim, the Office cites to column 4, lines 24-39 and column 5, lines 18-19. Applicant has studied Mutschler and can find no disclosure or suggestion of the subject matter of this claim. Accordingly, for at least this reason, this claim is allowable.

Claims 49-53 depend from claim 48 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 48, are neither disclosed nor suggested in the references of record, either singly or in combination with one another. In making out the rejections of these claims, the Office cites to various portions of Mutschler. Applicant has thoroughly studied Mutschler and, in particular, the excerpts cited by the Office, and can find no disclosure or suggestion whatsoever of the subject matter of these claims.

Claim 54 one or more computer-readable media having computer-readable instructions thereon which, when executed by a computer, cause the computer to:

- navigate to a network site that maintains at least one software application program;
- download a software application program comprising multiple different functionalities that can assist a user in accomplishing different tasks, the software application program being configured to be extended with software extensions that are deliverable via the network and described by at least one network-deliverable file; and
- ***extend the software application program by adding at least one extension to the application program, the extension being added by using a link to navigate to a different network site that hosts one or more files that describe the extension, and extension files that are used to implement the extension and downloading the one or more files and the extension files to a client.***

In making out the rejection of this claim, the Office cites to column 4, lines 24-26. Applicant has reviewed this excerpt and can find no disclosure or suggestion of the subject matter of this claim. Accordingly, for at least this reason, this claim is allowable.

Claim 55 recites a method of delivering software via a network comprising:

- ***accessing a Web site*** through which one or more software extensions can be obtained;
- ***receiving at least one file that describes at least one software extension using a hierarchical language that describes the software extension's logical attachment to a software application program***;
- receiving one or more software extension files; and
- installing the one or more software extension files based, at least in part, on the description contained in said at least one file.

In making out the rejection of this claim, the Office cites to column 6, lines 11-16 (as anticipating the "accessing" act), to column 4, lines 21-39 and column 6, lines 29-49 (as anticipating the "receiving" and "installing" acts). Applicant respectfully disagrees. The excerpts cited by the Office are set forth below in their entireties:

> The ORB 14 is coupled to the tool 15 by means of an import/export module 22; and, in a like manner to the tool 17 by means of an import/export module 23. The term "import" as used herein shall mean the creation of an object based on a description of an object transmitted from an external entity. *Column 6, lines 11-16.*
>
> A solution in accordance with the present invention employs the benefits of XMI (XML Metadata Interchange), which is an open industry standard that combines the benefits of the Web-based XML standard for defining, validating and sharing document formats on the Web with the Meta Object Framework (MOF) to provide a means for generating formats to allow the development tools to share information. One particular use of this invention is to define an XML DTD for the object-oriented Unified Modeling Language (UML). The XMI specification provides application developers with a common language for specifying transfer syntax for development language that allows visualizing, constructing and documenting of distributed objects and business models. The XMI specification in conjunction with the present invention will enable integration of development tools from multiple vendors, collaboration and distribution of object-oriented design and database schema information, and enhancement of the life cycle of information resources. *Column 4, lines 21-39.*

> Alternatively, the input/export module 22 or 23 can use the generated DTD to extract model information from the Tool 15 or the Tool 17, and to create an XMI data stream. This data stream can be communicated via the ORB 14 to the Repository Services 13, which can use the DTD to populate an application model such as the Model 21 in the Repository 11.
>
> There are various methods by which the DTD generator 19 can produce the DTD (bubble 19A). The method described herein produces a compact DTD, which allows one to group the various Attributes, Associations and Composition for later referential use. As the DTD productions in the first above-cited co-pending patent application (hereafter referred to as the "First Rule Set") are very simple, they can result in large DTD's. The repetition of detail also makes it difficult to perform modifications for the purposes of extension or experimentation. This is due to the fact that the object contents and any enumerated attribute list values are given for not only an object but for all of the Classes from which it is inherited, direct or indirect. *Column 6, lines 29-49.*

This subject matter neither anticipates nor renders obvious the subject matter of the present claim. Applicant can find no meaningful correlation between this subject matter in Mutschler and the subject matter of the present claim. Accordingly, for at least this reason, this claim is allowable.

Claims 56-62 depend from claim 55 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 55, are neither disclosed nor suggested in the references of record, either singly or in combination with one another. In making out the rejections of these claims, the Office cites to various portions of Mutschler. Applicant has thoroughly studied Mutschler and, in particular, the excerpts cited by the Office, and can find no disclosure or suggestion whatsoever of the subject matter of these claims.

Claim 63 recites a method of providing software for delivery over a network comprising:

- ***describing one or more software extensions using one or more extensible markup language (XML) files***, the extensions being configured for incorporation in a software program executing on a client;
- ***associating the one or more XML files with one or more associated extension files*** that are useable to provide a program functionality; and
- ***storing the XML files and associated extension files in a network-accessible location***.

Applicant has thoroughly studied Mutschler and can find no disclosure or suggestion of the subject matter of this claim. The Office contends that the subject matter of this claim is disclosed in column 4, lines 21-39. Such is simply not the case. Accordingly, for at least this reason, this claim is allowable.

Claim 64 recites a network site through which a client can access software files comprising:

- ***one or more software extension files*** configured to be incorporated into a software application program and delivered via a network; and
- ***one or more files associated with the one or more software extension files and describing the extension files***, the one or more files describing a logical attachment of the one or more software extension files to the software application program.

The Office contends that the subject matter of this claim is disclosed in column 6, lines 11-16, column 4, lines 21-39, and column 6, lines 29-49. These excerpts appear above and in no way anticipate or render obvious the subject matter of this claim. For at least this reason, this claim is allowable.

Claims 65 depends from claim 64 and is allowable as depending from an allowable base claim. This claim is also allowable for its own recited features which, in combination with those recited in claim 64, are neither disclosed nor suggested in the references of record, either singly or in combination with one another. In making out the rejection of this claim, the Office cites to portions of Mutschler. Applicant has thoroughly studied Mutschler and, in particular, the excerpts cited by the Office, and can find no disclosure or suggestion whatsoever of the subject matter of this claim.

Claim 66 recites a method of managing network-based software extensions comprising:

- ***grouping multiple software extension descriptions in a catalog*** in a network-accessible location;
- accessing the network-accessible location; and
- ***using the catalog to update a software extension*** that is resident on a computing device.

In making out the rejection of this claim, the Office cites to column 5, lines 16-23, to column 6, lines 11-12, and to column 6, lines 21-36. Applicant has reviewed these excerpts and can find no disclosure or suggestion of the subject matter of this claim. Accordingly, for at least this reason, this claim is allowable.

Claims 67-69 depend from claim 66 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 66, are neither disclosed nor suggested in the references of record, either singly or in combination with one another. In making out the rejections of these claims, the Office cites to various portions of Mutschler. Applicant has thoroughly studied Mutschler and, in particular, the excerpts cited by the Office, and can find no disclosure or suggestion whatsoever of the subject matter of these claims.

Conclusion

All of the claims are in condition for allowance. Accordingly, Applicant requests a Notice of Allowability be issued forthwith. If the Office's next anticipated action is to be anything other than issuance of a Notice of Allowability, Applicant respectfully requests a telephone call for the purpose of scheduling an interview.

Respectfully Submitted,

Dated: 8/6/03

By:
Lance R. Sadler
Reg. No. 38,605
(509) 324-9256